Low Level Design (LLD)

Credit Card Fraud Detection

Revision Number: 1.8

Last date of revision: 23/10/2021

Babita Pant

# Document Version Control

|  |  |  |  |
| --- | --- | --- | --- |
| Date Issued | Version | Description | Author |
| 21th sept 2021 | 1.1 | First Draft | Babita Pant |
| 30th sept 2021 | 1.2 | Added Workflow chart | Babita Pant |
| 5th Oct2021 | 1.3 | Added Exception Scenarios Overall, Constraints | Babita Pant |
| 9st Oct 2021 | 1.4 | Added KPIs | Babita Pant |
| 15th Oct 2021 | 1.5 | Added user I/O flowchart | Babita Pant |
| 17th Oct 2021 | 1.6 | Added EHR, LSTM model diagrams | Babita Pant |
| 21st May 2021 | 1.7 | Added dataset overview and updated user I/O flowchart. | Babita Pant |
| 23nd June 2021 | 1.8 | Restructure and reformat LLD | Babita Pant |
|  |  |  |  |

**Contents**

[Document Version Control 2](#_gjdgxs)

[**Abstract** 4](#_30j0zll)

[1](#_1fob9te) Introduction 5

[1.1](#_3znysh7) Why this Low-Level Design Document? 5

[1.2](#_2et92p0) Scope 6

[1.3](#_tyjcwt) Constraints 6

[1.4](#_3dy6vkm) 1.3 Risks 6

[1.5](#_1t3h5sf) 1.4 Out of Scope 7

[2](#_4d34og8) Technical specifications 7

[2.1 Predicting Disease 7](#_2s8eyo1)

[2.2 Logging 7](#_17dp8vu)

[2.3 Database 7](#_3rdcrjn)

[3](#_26in1rg) Technology stack 8

[4](#_lnxbz9) Proposed Solution 9

[5](#_35nkun2) Model training/validation workflow 10

[6](#_1ksv4uv) User I/O workflow 12

[7](#_44sinio) Exceptional scenarios 13

**Abstract**

We are living in an internet era in this 21st century. Now a-days online transaction has become a prominent mode of payment and credit card is one the most popular methods. But there is an apprehension be Victim of fraud as well. Recently we have seen so many fraudulent transactions through credit card. According to the dataset we have found 492 fraud Transactions out of 284,807 transactions in European Countries within two days. It is important that credit card companies must recognize fraudulent credit card transactions so that customers could not be charged for those items that they had not purchased.

To resolve this grievous problem we have tried to create a **Web Application** for the detection these types of frauds with the help of machine learning algorithm.

# Introduction

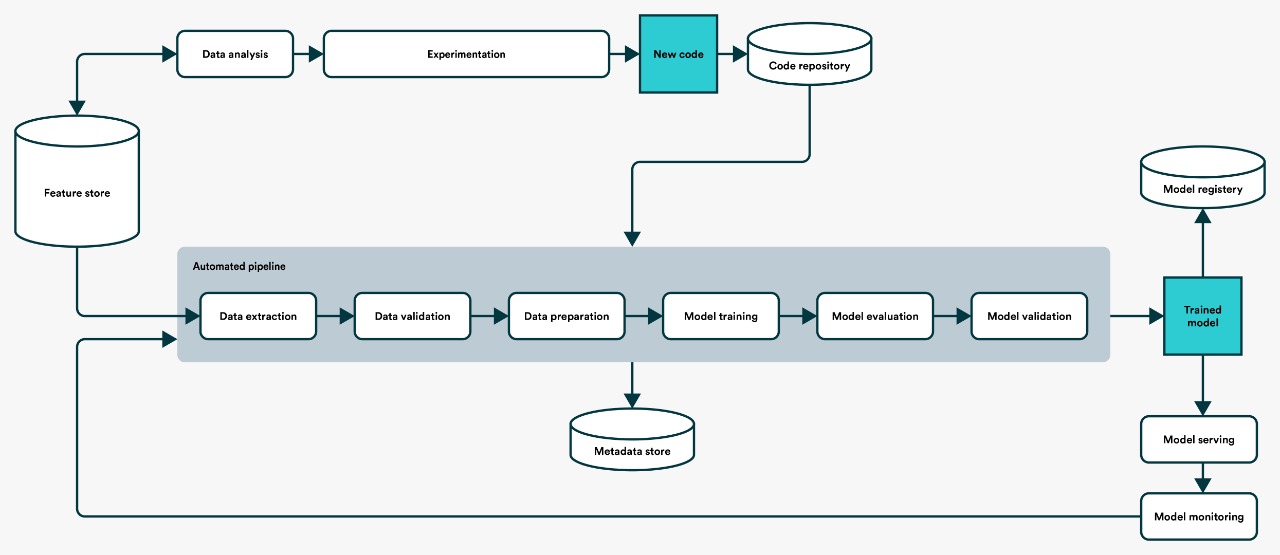
## Why this Low-Level Design Document?

The purpose of this document is to present a detailed description of the credit card fraud. It will explain the purpose and features of the system, the interfaces of the system, what the system will do, the constraints under which it must operate and how the system will react to external stimuli. This document is intended for both the stakeholders and the developers of the system and will be proposed to the higher management for its approval.

The main objective of the project is to predict credit card transaction is fraud or not.

Credit card fraud information, such as:

* Time
* V1 to V27 features
* Amount



## Scope

This software system will be a Web application this system will be designed to detect the fraud so that customers are not charged for items that they did not purchase.

## Risks

Document specific risks that have been identified or that should be considered.

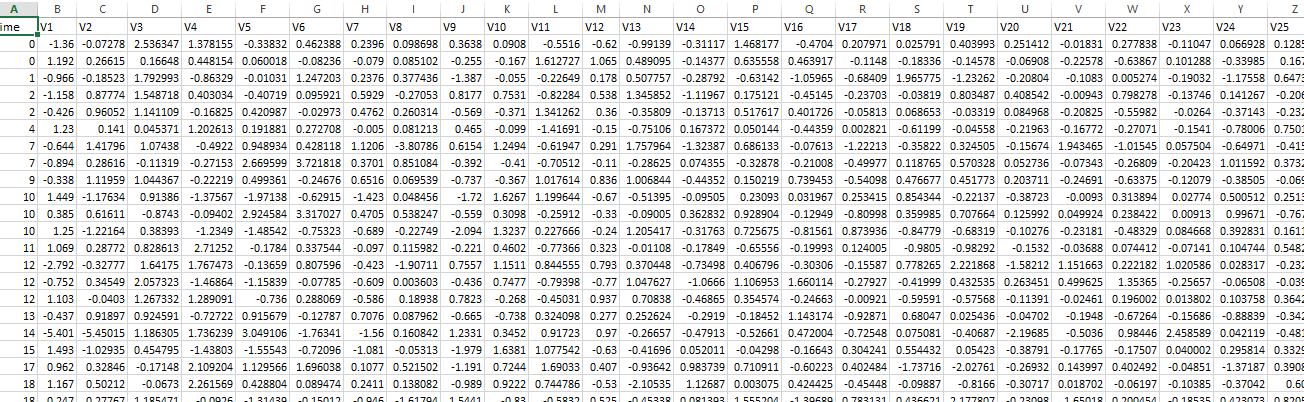
## Out of Scope

Delineate specific activities, capabilities, and items that are out of scope for the project.

## 2.1 Dataset

## 2.1.1 Credit Card dataset overview

Consists of 1 different table, table consists of the credit card information and most importantly we have the historic data of a credit card .Whereas the transcript table consists of credit card data.



## 2.1.2 Input schema

|  |  |  |
| --- | --- | --- |
| **Feature name** | **Datatype** | **Null/Required** |
| Time | float | Required |
| V1to V27 | float | Required |
| Time | float | Required |

## 2.2 Predicting Fraud

* The system displays fraud happened or not.
* The system presents the set of inputs required from the user.
* The user gives required information.
* The system should be able to predict whether fraud for the chosen transaction based on the user information.

## 2.3 Logging

We should be able to log every activity done by the user.

* The System identifies at what step logging required
* The System should be able to log each and every system flow.
* Developers can choose logging methods. You can choose database logging/ File logging as well.
* System should not be hung even after using so many loggings. Logging just because we can easily debug issues so logging is mandatory to do.

## 2.4 Database

System needs to store every request into the database and we need to store it in such a way that it is easy to retrain the model as well.

1. The User gives required information.

2. The system stores each and every data given by the user or received on request to the database. Database you can choose your own choice whether MongoDB/ MySQL/Cassandra.

**2.5 Deployment**

Heroku



# Technology stack

|  |  |
| --- | --- |
| **Front End** | HTML/CSS/ |
| **Backend** | Python Flask |
| **Database** | Cassandra |
| **Deployment** | Heroku |

Proposed Solution

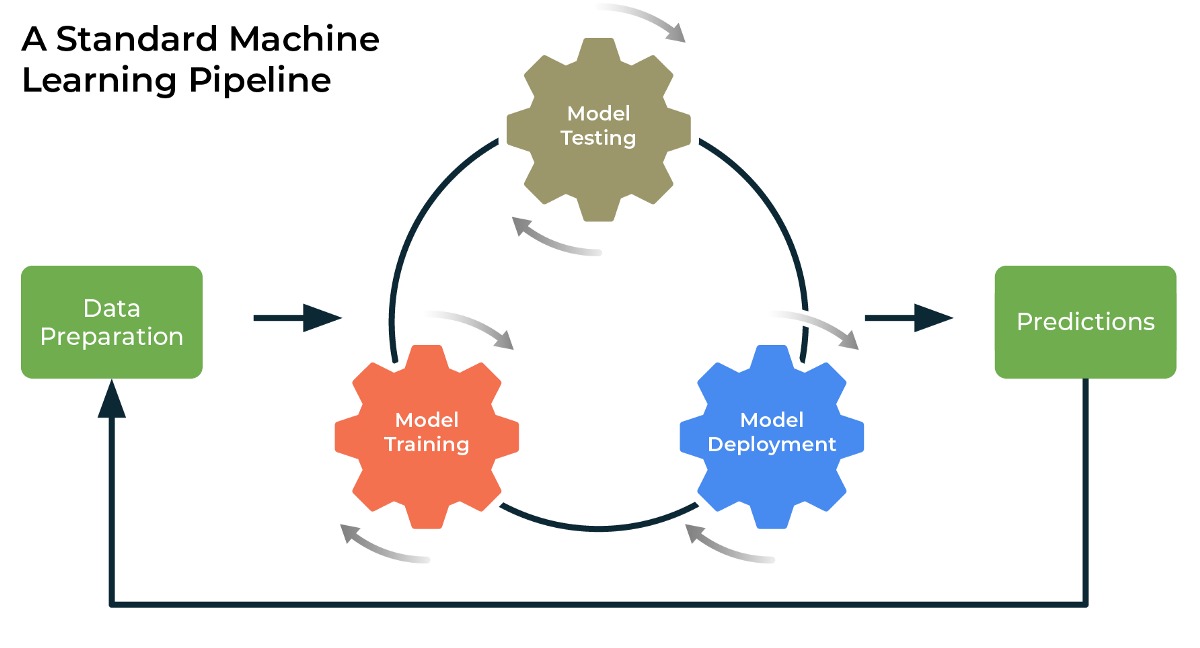
To prevent customer from fraud transaction.

1. Baseline Model: Random Forest, since this is a classification problem.
2. Actual model: Random Forest.

Model training/validation workflow



# User I/O workflow



# Exceptional scenarios

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Exception | Mitigation | Module |
| 21th Sept 2021 | 1.1 | First Draft | Babita Pant |
| 23rd Oct 2021 | 1.2 | Last Draft- Added Workflow chart | Babita Pant |

# Key performance indicators (KPI)

* Time and workload reduction using the Random Forest model.
* Comparison of accuracy of model prediction.